PATENT
PD-00-1004

EXTENSIBLE FIRMWARE INTERFACE VIRUS SCAN

Jonathan T. Huntington II
Richard A. Bramley

EXTENSIBLE FIRMWARE INTERFACE VIRUS SCAN

BACKGROUND

The present invention relates generally to computer systems and methods, and more particularly, to methods that provide protection from software viruses on computer systems that use an Extensible Firmware Interface.

Viruses are code that is hidden or contained in a file or software application that, when processed on a receiving computer, causes problematic activity on the receiving computer. The virus may take many forms, such as one that is part of a file attached to an E-mail message, which when opened, activates embedded code that creates havoc on the receiving computer or causes the receiving computer to initiate unwanted activities.

Virus protection is a large business and there are a number or organizations whose primary goal is to detect and produce anti-virus software that thwarts existing and new viruses. These organizations include Symantec Corporation and McAfee, for example. These organizations are dedicated to virus protection and remediation, and market various programs that provide for virus protection on various computer platforms.

The programs developed by these organizations operate to scan hard disk drives and scan incoming E-mail files and files input via floppy disks or other media to detect and eradicate the virus from the computer. The programs developed by these organizations work with PCs on DOS, Windows and Unix Operating systems, and other platforms and operating systems as well.

The following US Patents were uncovered in a search of the US Patent and Trademark Office patent database records by the present inventors that address software virus protection: US Patent No. 5,764,889, issued June 9, 1998 entitled "Method and apparatus for creating a security environment for a user task in a client/server system"; US Patent No. 5,032,979, issued July 16, 1991 entitled "Distributed security auditing subsystem for an operating system"; US Patent No. 5,815,573, issued September 29, 1998 entitled "Cryptographic key recovery system"; US Patent No. 5,796,830, issued August 18, 1998 entitled "Interoperable cryptographic key recovery system"; US Patent No. 5,590,266, issued December 31, 1999 entitled "Integrity mechanism for data transfer in a windowing system"; US Patent No. 4,918,653, issued April 17, 1990 entitled "Trusted path mechanism for an operating system"; and US Patent No. 6,026,374, issued April 17, 2000 entitled "System and method for generating trusted descriptions of information products". These patents address conventional mechanisms for providing virus protection.

A new Industry standard ROM based operating system has been developed which is known as EFI, or the Extensible Firmware Interface, that operates to replace DOS (Disk Operating System) functionality. As such, the EFI is controlled by the basic input and output system (BIOS) of the computer. The EFI is part of the BIOS within a flash nonvolatile RAM, and it is guaranteed to execute before any other operating systems are loaded or disk access is allowed. Virus protection is not generally available for this ROM based operating system. The present invention addresses this need.

It is an objective of the present invention to provide for a virus protection method (software or firmware) for use with computer systems employing the Extensible Firmware Interface.

SUMMARY OF THE INVENTION

To accomplish the above and other objectives, the present invention provides for a secure method (implemented as software or firmware) for implementing virus protection on a computer system comprising an Extensible Firmware Interface and a basic input and output system (BIOS). The method is designed to protect and remedy potential viruses.

The computer system includes a central processing unit, a hard disk, and a nonvolatile random access memory, such as a read-only-memory or flash memory device. The Extensible Firmware Interface is a ROM-based operating system (i.e., stored in the read-only-memory or a flash random access memory) that provides disk

operating system (DOS) functionality for the computer system, and is controlled by the BIOS.

The Extensible Firmware Interface is a read-only-memory (ROM) based operating system that operates to replace traditional disk operating system (DOS) functionality. The Extensible Firmware Interface is controlled by the basic input and output system and executes before any other operating systems are loaded or disk access is allowed.

The present method comprises the following steps. A command shell of the Extensible Firmware Interface is modified to include a command that operates to copy the boot sector of the hard disk to the nonvolatile random access memory. The modified Extensible Firmware Interface is stored in the nonvolatile random access memory. When the computer system is initialized (booted), a boot sector of the hard disk is copied to the nonvolatile random access memory. The boot sector of the hard disk is automatically read back from the nonvolatile random access memory on each boot, which bypasses the boot sector access of the hard disk during system initialization.

An extra field may be added to a BIOS SETUP routine, which is part of the BIOS, that allows a user to enable or disable reading the boot record from nonvolatile random access memory on boot. In implementing this aspect of the present method, the BIOS SETUP routine is run, and the user is prompted to enable or disable reading the boot record from nonvolatile random access memory on boot. The use of the BIOS SETUP routine allows a user to recover if he or she changes the boot disk or intentionally changes the boot disk's boot record to change the operating system or partition of the hard disk.

The method may also be modified to require entry of a security signature to prevent unauthorized updating of the stored boot sector. In implementing this aspect of the present invention, the command shell of the Extensible Firmware Interface is modified to include a command a security signature input field. At the appropriate time during execution of the Extensible Firmware Interface the security signature input field is displayed to a user. The required signature is then input by the user prior to updating the stored boot sector.

BRIEF DESCRIPTION OF THE DRAWINGS

The various features and advantages of the present invention may be more readily understood with reference to the following detailed description taken in conjunction with the accompanying drawing, wherein like reference numerals designate like structural elements, and in which:

Fig. 1 illustrates a first embodiment of an exemplary computer system in which the present invention is employed;

Fig. 1a illustrates a second embodiment of an exemplary computer system in which the present invention is employed; and

Fig. 2 is a flow diagram that illustrates an exemplary method in accordance with the principles of the present invention for providing virus protection of a computer system.

DETAILED DESCRIPTION

Referring to the drawing figures, Fig. 1 illustrates a first embodiment of an exemplary computer system 10 in which the present invention is employed. The first embodiment represents a typical older computer system 10 constructed in accordance with the current state of the art.

The computer system 10 comprises a central processing unit (CPU) 11, which is coupled to a hard disk 12, a read-only-memory (ROM) 13, and a nonvolatile random access memory (NVRAM) 14, also known as flash memory 14. The computer system 10 also comprises an Extensible Firmware Interface (EFI) 15 which is a ROM-based operating system (i.e., stored in the read-only-memory 13) that provides disk operating system (DOS) functionality for the computer system 10, along with a basic input and output system (BIOS) 16.

In general, ROM and flash devices are considered by BIOS engineers to be substantially the same. Historically, the BIOS was located in ROM but, in modern computer systems, the BIOS and EFI are contained in flash devices, so there is no need for the distinction between ROM and NVRAM (flash) devices. It is to be understood that the present invention does not exclude ROM devices, although substantially all currently produced personal computers are implemented using flash devices. A more current embodiment of the computer system 10 is discussed with reference to Fig. 1a.

The Extensible Firmware Interface 15 comprises a command shell, which is the outermost layer or user interface of this program, and which comprises a command processor interface. The command processor is a program that executes operating system commands. The command shell is that part of the command processor that accepts commands. After verifying that the commands are valid, the shell sends them to another part of the command processor to be executed.

The basic input and output system 16, or BIOS 16, is a firmware program that is stored in the nonvolatile random access memory 14 (or flash memory 14). The BIOS 16 brings up the computer system 10 when it is turned on. The Extensible Firmware

Interface 15 is controlled by the BIOS 16 and executes before any other operating systems are loaded or access is allowed to the hard disk 12.

The BIOS 16 determines what the computer can do without accessing programs from the hard disk 12 or other media. The BIOS 16 contains code required to control the keyboard, display screen, disk drives, serial communications, for example, along certain other functions, depending upon the computer system 10.

Fig. 1a illustrates a second embodiment of an exemplary computer system 10 in which the present invention is employed. The second embodiment represents a computer system 10 constructed in accordance with the current state of the art.

The second embodiment of the computer system 10 comprises a central processing unit (CPU) 11, which is coupled to a hard disk 12, and a nonvolatile random access memory (NVRAM) 14, or flash memory 14. The computer system 10 also comprises an Extensible Firmware Interface (EFI) 15 and a basic input and output system (BIOS) 16 stored in the NVRAM 14. The Extensible Firmware Interface 15 and BIOS 16 function as discussed with reference to Fig. 1.

Fig. 2 is a flow diagram that illustrates an exemplary method 20 in accordance with the principles of the present invention for providing virus protection for the computer system 10. The method 20 is designed to protect and remedy potential viruses that are loaded onto the computer system 10.

The method 20 comprises software 20, and preferably firmware 20, that is used in conjunction with a computer system 10 comprising a central processing unit (CPU) 11, a hard disk 12, a nonvolatile memory (NVRAM) 14, a basic input and output system (BIOS) 16, and an Extensible Firmware Interface 15. The software 20 or firmware 20 stored and executed from the nonvolatile memory (NVRAM) 14 or ROM 13) of the computer system 10. The method 20 comprises the following steps.

A command shell of the Extensible Firmware Interface 15 is modified 21 to include a command, referred to as <saveboot>, that operates to copy the boot sector of the hard disk 12 to the nonvolatile random access memory 14. The modified Extensible Firmware Interface 15 is stored 22 in the nonvolatile random access memory 14.

When the computer system 10 is initialized (booted), a boot sector of the hard disk 12 is copied 23 to the nonvolatile random access memory 14. The boot sector of the hard disk 12 is automatically read back 24 from the nonvolatile random access memory 14 on each boot, which bypasses the boot sector access of the hard disk 12 during system initialization.

An extra field may be added 25 to a BIOS SETUP routine 17, which is part of the BIOS 16, that allows a user to enable or disable reading the boot record from nonvolatile random access memory 14 on boot. In implementing this aspect of the

present method 20, the BIOS SETUP portion of the BIOS is run 26, and the user is prompted to enable 27 or disable 28 reading the boot record from nonvolatile random access memory 14 on boot. The use of the BIOS SETUP routine 17 allows a user to recover if he or she changes the boot disk or intentionally changes the boot disk's boot record to change the operating system or partition of the hard disk 12.

The software 20 or firmware 20 that implements the method 20 may also be modified to require entry of a security signature to prevent unauthorized updating of the stored boot sector. In implementing this aspect of the present invention, the command shell of the Extensible Firmware Interface 15 is further modified 31 to include a command a security signature input field. At the appropriate time during execution of the Extensible Firmware Interface 15 the security signature input field is displayed 32 to a user. The required signature is then input 33 by the user prior to updating the stored boot sector.

The method 20 is fast because the operating system does not need to scan the boot sector for a long list of potential viruses during power on self test (POST). The method 20 is simple because it does not require any additional virus software. In addition to boot sector protection, additional software may be provided during the power on self test to scan for infection signatures in the nonvolatile random access memory 14 on each boot. Signature files may be provided in firmware or the nonvolatile random access memory 14 or from the hard disk 11. Remediation code may also be provided to remove infected boot sectors and viruses found in memory during power on self test. The present virus protection method may also replace or complement other virus protection programs.

Thus, a method that provides protection from software viruses on computer systems that use an extensible firmware interface has been disclosed. It is to be understood that the above-described embodiments are merely illustrative of some of the many specific embodiments that represent applications of the principles of the present invention. Clearly, numerous and other arrangements can be readily devised by those skilled in the art without departing from the scope of the invention.